Utah Master Directory

Design Document

prepared for **State of Utah**

www.novell.com

Date 06-17-02

Novell®

Prepared By

David R. Lee dlee@novell.com

Deron Tinsley dtinsley@novell.com

Stuart Proffitt sproffitt@novell.com

Details

Title:	Document Type
Version:	1.0
Status:	Fanal
Creation:	05-23-02
Delivery:	06-17-02

Project Number 285280

Contents

Table of Figures

Table of Tables

Executive Summary

State of Utah Objectives

Almost every time a new application is added into an environment, like the State of Utah's, an administrator must create accounts for the users of the new system, and users must remember new names and passwords to access the new system. By creating an Enterprise Authentication Directory (Utah Master Directory [UMD]) the State hopes to alleviate these administrative and user issues. The State of Utah would also like to automate the creation of user accounts in UMD by having the HR system create new employees in UMD on their hire date. This will allow existing and future LDAP applications to use UMD for user authentication, authorization, and profiling. This directory will also be used to provision accounts to other eDirectory resource trees (UTSTATE, UDOT, UTCOURTS, and USOR), messaging systems such as GroupWise or Exchange, and other applications/databases/directories such as Active Directory. In the future provisioning within the UMD may extend to the state's phone systems, Security Badge systems, and eREP. In the following diagram, UMD is represented by the shaded triangle below.

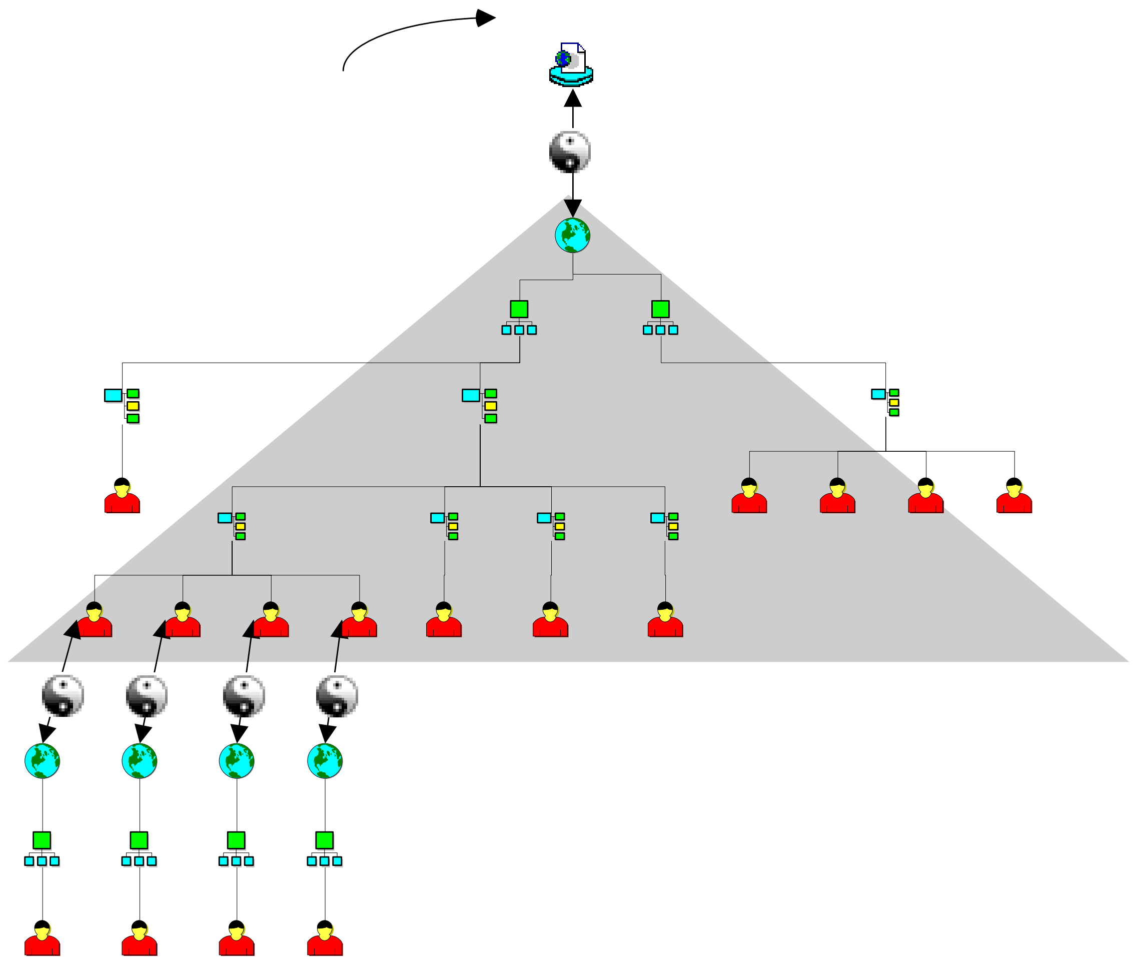

Figure 1: Utah Master Directory Overview

This engagement will focus on creating and synchronizing data for employee accounts from the HR system to the Utah Master Directory and then, depending on the employees OrgID (a combination of Agency ID and low org), the user will be created in the appropriate resource directory tree(UTSTATE, UDOT, UTCOURTS, and USOR.) An email will be sent to the employee's manager and OrgID's Distribution List (which will contain the local LAN administration team) informing them that a new user has been created. It will also inform them of the random password that has been assigned to the user and a suggested unique email for them to create until the automatic provisioning of email has been accomplished.

When the new employee email has been created in GroupWise, the email address will be synchronized to UMD and then to HR so as to allow HR to maintain consistent information. When a phone number has been assigned to the user, this information will also be sent to HR. When an employee has information updated in HR, the information will be sent to UMD and then to the appropriate subscribers of UMD. This data synchronization will occur automatically based upon business processes that will be discussed later in this document.

Pub

While the creation and updating of information is a valuable feature of this solution, the most powerful feature is the security aspect. For example, when an employee is terminated all associated resource tree accounts will be disabled. The employee's user object will be moved to the UMD Public container, and their access to State of Utah employee applications and web pages that use UMD for authentication will be removed.

The UMD tree will become an integral part of the State of Utah's future authentication and identity management processes.

DirXML General Information

DirXML Overview

DirXML is an event driven utility that provides flexible synchronization between disparate systems so that enterprise data can be kept up to date with minimal maintenance costs. DirXML is one technology that will be used for the Utah Master Directory (UMD) Project. Using DirXML, the State will synchronize user account information from the HR system to the UMD tree. While DirXML drivers function out of the box, most need to be configured to support business processes. Part of this configuration identifying required attributes, authoritative sources, object matching rules, data transformations, and event transformations and the mapping of schemas,.

Schema Mapping

Since applications have different ways of representing employees, users, or mailboxes within their systems, DirXML needs a method to match the Object Classes (accounts or records) and Attributes (fields) between the different applications. This process is known as schema mapping. By performing a schema mapping, the underlying applications do not need to be modified to match eDirectory. Each application will have a driver that will have a mapping between its' object classes and attributes so as to allow the DirXML configuration to be based on eDirectory attributes. This enables information to come into UMD and be sent out to each application, without requiring applications to know about any other application's schema.

Required Fields and Authoritative Sources

Applications may have required fields that must be filled in before an account can be created in the application's system. The State's business processes require specific information to be obtained before an account is considered valid. By identifying these required attributes for both the application and the business logic, DirXML can enforce the business processes by "vetoing" the creation of an account until all required attributes have values.

The HR system is considered the authoritative source for employee accounts at the State of Utah. Each email system is considered authoritative over the Internet EMail Address. Filters are used to determine which attributes are allowed to flow between HR, the UMD and the Resource Trees (UTSTATE, UTCOURTS, UDOT, USOR). Filters are also used to control the direction of data flow, since each application can change information that is stored in its system. While the business processes at the State do not allow changes to the Employee Identification Number (workforceID) in UMD, there is nothing stopping a "rogue" administer from making changes in their own Resource Tree. And as such, the State is encouraged to create business rules that strongly discourage this type of behavior.

Object Mappings

Since some applications might have existing users/accounts there must be a method to match objects between the systems. DirXML uses a process known as Object Mapping to allow existing objects to be matched to existing or newly created eDirectory objects. This match is based upon unique matching criteria. When an exact match can be determined, the objects are associated.

For the Utah Master Directory, it has been identified that an Employee's identification number (workforceID) is the easiest way to uniquely identify objects as being the same between the HR, UMD and the resource trees. HR is considered to be the authoritative source of creating new workforceIDs. If a user is created in a resource tree prior to having a workforceID assigned, a match will only be made between the resource tree and UMD if there is a match on the user's last name (Surname) and workforceID. As an added security measure, the creation of an association will require the presence of the resource tree's specific auxiliary class on the UMD user object

Data Transformations

Different applications might not represent data in the same format or method depending on the applications needs. For this reason, DirXML allows for data transformations. Each data element can be manipulated from one system to the next into the format desired in that system. This can be as simple as translating a full name into three attributes - Given Name, Middle Initial, Last Name or as complex as applying an external process to a series of attributes. It has been determined that the UMD solution will have limited data transformations (i.e. Phone numbers will be in the xxx xxx-xxxx format , the middle name will be the first eight (8) characters of the HR field empl_second_name, and the full name will be constructed from the Given Name and Surname.)

Event Transformations

Not only can data in the applications be represented differently, but the State's business process events can also be transformed. DirXML supports Add, Modify, Rename, Move and Delete events. Each event can be transformed into another event, blocked or simply passed through. For example, when a user is terminated in HR, the user will not be deleted in the resource trees but disabled instead. DirXML will only see events that occur on the server that is hosting the DirXML Engine. For this reason a replica of each partition that needs to be synchronized must exist on these servers.

DirXML Fail-Over

DirXML guarantees that events that occur will get to the destination application. This is done through the use of an event cache to store events waiting to be processed. This cache is stored on the same volume that eDirectory is installed on. As long as this volume does not become unavailable due to a hardware failure, a restart of the server is typically the best fail-over plan. Additional fail-over design can be provided if needed.

Monitoring the DirXML Drivers

The DirXML Drivers create an entry in the dirxml.log file (SYS:SYSTEM\DIRXML.LOG) when errors occur. Typical errors relate to LAN issues or other communication issues. This file should be monitored regularly. Errors encountered during development and testing will be documented in the Support documentation for quicker resolution. Beyond communication errors the only other errors expected are when a "rogue" administrator defies the business logic that the drivers were created for. Some of these errors will be documented in the Support documents but not all occurrences can be realized. For this reason it is recommended that the support personnel from the State become familiar with the logic configured for the DirXML drivers. ConsoleOne also provides an Application Driver Monitor that will show the current status of a driver set.

Overall Design Guidelines for:

HR - Utah Master Directory (UMD)

The following overall design guidelines were followed:

- The HR Sybase system is the authoritative source of all employees
- When an employee is hired, and synchronized to UMD, they will receive a suggested email account
- An employee's OrgID will determine:
    - What Internet email domain name will be used to create an email name (i.e. @utah.gov, @email.utcourts.gov, …)
    - What email post office the user's mail will be stored
    - What domain the post office is in
    - What resource tree the user will be created in
        - Unique emp_agy have the same res_tree
    - Where the user will be placed in the resource tree
    - Which template object the user will be created with
    - Which Distribution List to send email to for certain events
- empl_org is not unique between agencies but unique within an agency
- When an employee is created in the Utah Master Directory (UMD) a random password will be assigned that must be changed on the next login of the user
- HR will insert a new record into dbo.umd_empl_master when a new employee is hired. This record will also pull the associated OrgID information from dbo.umd_resource (as documented above)
- There is an empl_status field (mapped to UTIW-Status) that will have the following statuses:
    - A Active
    - I In-Active (LWoP)

(Disable user in downstream resource trees and not allow

access to some employee web pages)

- T Terminate

  (Disable/delete user in downstream resource trees)

- N New Hire

  (Do not create user in downstream resource trees by setting empl_nw_access to false)

- The empl_status will change from N to A on the effective date of the hire
- The empl_nw_access will disable all accounts in connected trees. SiteMinder and other security tools should look to this attribute to determine if the account should be allowed access to certain applications
- There is a protected_record_flag, UTIW-Invisible, that when set will cause the employee to be created in a protected container.
- When an employee's OrgID changes an email will be sent to the previous and the new Distribution lists.
- An environmental variable will exist (v_EmailOnTransfers) that will prevent emails from being sent on changes of OrgID for the two periods of time that mass changes occur on OrgID. This variable must be set previous to HR making the mass change and once the changes have occurred, setting it back for further emails to occur.

Utah Master Directory (UMD) to Uxxx (UTSTATE/UDOT/USOR)

The following overall design guidelines were followed:

- Changes in UMD will be done through approved interfaces following the guidelines specified below.
- Agencies will never change Primary Resource Trees without changing emp_agy (UTIW-AgencyNumber)
- Mass updates of OrgIDs will cause significant emails unless the v_EmailOnTransfers is set to False in the driver(s) XSLT stylesheets.

  Note: failure to return this value to true will result in no new UTIW-LowOrg transfer notifications when the placement does not change.

UMD eDirectory Design

Design Specifications

The design of the UMD eDirectory tree is a relatively "flat" design. While the initial scope of this project is to create and Enterprise Authentication Directory for approximately 25,000 employees, a future phase will be to expand the user population to include all citizens in the State of Utah, persons who conduct business with the State of Utah (reserve a park, contract with the state, ….), and any other people who use services that the State of Utah provides. This means that the tree could easily expand to several million users. DirXML will be used by The UMD tree to provision government employees to several eDirectory resource trees, and may eventually expand to include provisioning to Active Directory, GroupWise, Exchange or other systems that require user accounts that can not authenticate via LDAP. The UMD tree will also be used for authentication, authorization and profiling by Netegrity's SiteMinder and Novell's Portal Services. Future provisioning applications (Novell's eGuide or similar) may also be used to allow for the future goal of employee self-provisioning.

Business Objectives

The purpose of the UMD eDirectory tree is to play the role of the Enterprise Authentication eDirectory tree for Government and Public users.

- Design to expand from 25,000 to 3,000,000 objects
- LDAP authentication
- DirXML Provisioning

Tree Structure

eDirectory is a fault tolerant, highly scalable, flexible directory service. The design of the UMD eDirectory tree is intended to facilitate the defined business processes of the State of Utah. The diagram below is a view of the UMD eDirectory tree with partition boundaries.

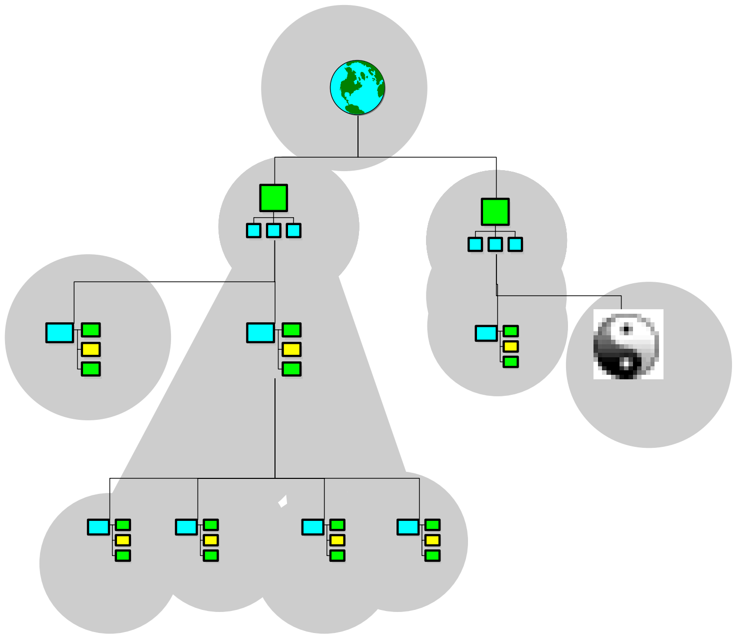

Figure 2: UMD with partitions

Container explanation

The following describes the use of each of the eDirectory containers within the UMD tree:

IS:
UMD will use this container to hold services required for the following containers (a container is a logical placeholder similar to a directory on a file system) and to hold system related user accounts:

- OU=Trees.O=IS: This container will hold a Tree Admin user object per resource tree that will be used by each DirXML Driver for authentication and authorization purposes. Each user will also contain the Distribution List to email problems for that tree.
- OU=DirXML Driver Set.O=IS: This container will hold all of the DirXML Drivers.

Gov:
UMD will use the following containers to hold Government related user accounts:

- OU=Emp.OU=Gov.O=UT: This container will hold Utah employees that are in HR.
- OU=Protect.OU=Gov.O=UT: This container will hold Utah employees that are in HR and have the protected_record_flag set.
- OU=Hold.OU=Gov.O=UT: This container will hold email addresses for employees that have been terminated. This will allow the JDBC driver to not suggest an email address of a user who has been terminated as the State has requested a 90 day no re-use policy of addresses in case the user rehires.

- OU=Temp.OU=Gov.O=UT: This container will hold Contractors, Vendors and Volunteers who work with the State. Currently a process has not been defined to populate this container as HR does not hold these types of users.

Pub:
UMD will use this container to hold user accounts for the Citizens of Utah and all users who will have a need to use services that the State of Utah provides. There will be 10 sub-containers (0...9) where public users will be created depending on the last digit of their CN.

Schema

The schema of a directory defines the types of objects and the data that they can contain. By default, eDirectory offers a base schema that is fully functional and defines a wide variety of objects and their attributes.

A detailed list of the defined object classes and their associated attributes can be found in Appendix A that are involved with this statement of work.

Each object class has a specific purpose in the eDirectory tree. Objects should be created using approved methods to ensure that all objects are associated to the appropriate object classes.

User Templates

User Templates will be used to create users in most of the resource trees. The exact template for each low org will be stated in the HR system. The State of Utah is encouraged to maximize the use of user templates in each resource tree, as this will greatly simplify the user object creation process for the resource tree administrators.

Indexing

Indexes are created on a server-by-server basis to facilitate searching the eDirectory tree. These indexes sort the data contained in the directory by the attribute that is indexed. This increases the speed of a search performed on that attribute by an order of magnitude. The following attributes are indexed by default in eDirectory:

- Aliased Object Name
- CN
- CN_SS
- dc
- Equivalent To Me
- Given Name
- Member
- Obituary
- Reference
- Surname

- uniqueID
- uniqueID_SS

Additional indexes will be needed for SiteMinder to quickly lookup information for login purposes:

- Internet EMail Address
- UTIW-NWAccess
- workforceID
- UTIW-EmpEmail
- UTIW-PublicEmail

Background Processes

eDirectory is a reliable, scaleable, and fault tolerant directory. Its default configuration provides full functionality for the majority of customers. All implementations can benefit from modifications and tuning to the background processes to provide better performance. The following sections detail the design of background processes of UMD.

Partitioning

Partitions are logical divisions of the eDirectory database that forms a distinct unit of data in the eDirectory tree that administrators use to store and replicate eDirectory information. Each partition consists of a container object, all objects contained in it, and the information about those objects. Partitions do not include any information about the file system or the directories and files contained there.

A partition is generally created to divide the data contained in the eDirectory database into logical units. Partitioning a tree can decrease the amount of data synchronized to a server, allowing eDirectory to perform better across a slow network connection (i.e. WAN Link).

Partitioning the eDirectory also allows an administrator to compartmentalize data. This process facilitates the repair of a section of the directory, while allowing the remainder of the directory to continue to function as normal.

The UMD eDirectory tree will be partitioned, as shown in Figure 2, at the following locations:

- [Root]
- UT
- Pub.UT
- IS
- DirXML Driver Set.IS

Replication

Instead of storing a copy of the entire eDirectory database on each server, you can make a copy of the eDirectory partition and store it on many servers across the network. Each copy of the partition is known as a replica. You can create any number of replicas for each eDirectory partition and store them on any server. The types of replicas include master, read/write, read only, subordinate references, filtered read/write, and filtered read only. The following table describes the replica types:

Master, read/write, and read only replicas	Contain all objects and attributes for a particular partition.
Subordinate references	Are used for tree connectivity.
Filtered replicas	Contain a subset of information from the entire partition, consisting of only the desired classes and attributes. Desired classes and attributes are defined by the server's replication filter, which is used to identify which classes and attributes are allowed to pass during inbound synchronization and local changes. Filtered replicas are **NOT** being used in the UMD Tree

A server may contain a copy of all partitions in the directory tree, or a subset of those partitions. Each copy of a partition is referred to as a replica. Replication is the process in which all replicas are updated with information that has changed. A change can take place on any Read/Write replica (or copy) of a partition. That replica will then propagate those changes to the other copies of a partition thru the replication process. eDirectory automatically configures and manages replication of a directory tree.

The following diagram represents the replica matrix of the UMD eDirectory tree.

Partition Name	**[ROOT]**	**UT**	**Pub.UT**	**IS**	**DirXML Driver Set.IS**
Server Name					
UTAH1	M	M	M	M	M
UTAH2	R/W	R/W	R/W	R/W	R/W
UTAH3	R/W	R/W	R/W	R/W	R/W

Time Synchronization

Time synchronization is a service that maintains consistent server time across the network. Time synchronization is provided by the server operating system, not by eDirectory. eDirectory maintains its own internal time to ensure the proper order of directory updates, but it gets its time from the server operating system.

The method used to synchronize time on each server will vary by the server operating system. At the State of Utah, the time synchronization design in the UMD eDirectory tree will be based on NTP with all servers designated as "Secondary" and getting their time from an multiple NTP data sources that are all synchronized to the same Stratum 1 server.

Novell NetWare

In IP networks, NetWare 6.x servers communicate time with other servers using IP. NetWare 6.x servers use TIMESYNC.NLM and Network Time Protocol (NTP) to accomplish this. TIMESYNC.NLM on NetWare can be configured as a NTP provider, consumer, or both.

The NetWare servers in the UMD eDirectory tree will have TIMESYNC.NLM configured to consume time from two NTP time sources. The TIMESYNC.CFG for the Secondary timeservers (NetWare 6.x) resembles the information below.

```
# TIMESYNC Configuration Parameters

Configured Sources =    ON
Directory Tree Mode =   ON
Hardware Clock =        OFF
Polling Count =  3
Polling Interval =      600
Service Advertising =   OFF
Synchronization Radius =        2000
Type =  SECONDARY

# TIMESYNC Configured time source list

TIME SOURCE = NTP1:123

TIME SOURCE = NTP2:123
```

Service Location Protocol

Novell eDirectory servers, to determine the network address of other servers over IP, use Service Location Protocol (SLP). These network addresses are used by eDirectory to facilitate data synchronization, time synchronization, and internal eDirectory communication. SLP design is an important part of the design of an eDirectory Tree. Service Location Protocol (SLP) provides a similar function in IP networks as SAP provides in IPX networks. It registers information in a database and allows clients to query the database to find services. Unlike SAP, SLP has some properties that make it a more attractive service location protocol:

- SLP is a passive protocol, which is unlike the active SAP protocol. SAP forces itself out on the wire once per minute to advertices services. SLP is found on the wire only when user agents need to retrieve a list of available services on the network.
- SLP is a pull technology, whereas SAP is a push technology. Pull technologies lend themselves better to the tailoring and customization of queries than push technologies. This provides for much more efficient use of network bandwidth because a pull technology like SLP places only useful information on the wire. A push technology like SAP tries to place all information needed for any query on the wire, whether it is truly needed or not.

But this better technology comes at a cost. That cost is either an administrative cost or a network bandwidth cost. SLP can use different types of TCP/IP packets to facilitate a service resolution infrastructure. There are three types of packets in TCP/IP that can be used for communication between hosts. They are as follows:

Unicast	This is a directed packet between two hosts on the network. The packet is addressed specifically to the destination host. All other hosts on the network that see the packet ignore it.
Broadcast	This is a general packet destined for all hosts on the same network as the sending host. Every host on the same network must process the packet. Because CPU cycles must be spent processing the packets coming into the host, broadcast storms can occur and affect machine performance.
Multicast	This packet is designed to reach a group of hosts on the network. There is a reserved range of IP addresses whose first octet ranges from 224 to 239. These addresses are multicast addresses and are a way to logically identify a group of hosts on the network. Hosts on the network join the group upon booting, and when a packet is addressed to a specific multicast address, all members of the multicast group receive the packet. Non-members that see the packet ignore it.

For SLP to work without any administrative overhead as SAP currently does, multicast must be enabled on the network to facilitate communications between user and service agents. If multicast is not an option, then a directory agent must be configured for the network, and the service agents on the network must be configured to find the directory agent.

Through Novell's integration of SLP with eDirectory, local SLP information is compiled to provide a global representation of all available services on the network. This provides dynamic discovery of services locally and scalability in large networks.

The following objects represent SLP resources in eDirectory:

- SLP Service object
- SLP Directory Agent object
- SLP Scope Unit

For the UMD eDirectory tree, the SLP configuration consists of the following:

- SLP Directory Agent object = .CN=UTAH1???.O=IS
- SLP Scope Unit object = .CN=Global.O=IS
- Scope Name =DAS-SCOPE

Prior to customizing the configuration of servers all SLP eDirectory objects must be created. The following objects should be created with the associated information:

Object Name	Object Type	Attributes Configured
.CN=UTAHDA.O=IS	SLP Directory Agent	Serviced Scope Units = .CN=DAS-SCOPEO=IS Host Server = .CN=UTAH1.O=IS
.CN=DAS-SCOPE.O=IS	SLP Scope Unit	Scope Name = DAS-SCOPE Filters = (no filters set) Scope Authority = (no Scope Authority set)

Management Accounts

Beyond the normal administrative accounts DirXML should have an account to be equivalent to so as to track modifications made by the driver processes. The following is a list of administrative and service accounts to be used in the UMD tree:

Account Name	Purpose	Rights Assigned
Admin	Administer the tree	Full administrator privileges (Entry Rights SBCRD) to [Root]
UTSTATE	Account used by DirXML for directory management for UTSTATE Driver	Full administrator privileges (Entry Rights SBCRD) to [Root]
UTCOURTS	Account used by DirXML for directory management for UTCOURTS Driver	Full administrator privileges (Entry Rights SBCRD) to [Root]
UDOT	Account used by DirXML for directory management for UDOT Driver	Full administrator privileges (Entry Rights SBCRD) to [Root]
USOR	Account used by DirXML for directory management for USOR Driver	Full administrator privileges (Entry Rights SBCRD) to [Root]
USOE	Account used by DirXML for directory management for USOE Driver (future)	Full administrator privileges (Entry Rights SBCRD) to [Root]
UTAHGOV	Account used by DirXML for directory management for Utah.Gov GW (Future)	Full administrator privileges (Entry Rights SBCRD) to [Root]

Hardware Configuration

The following information is the recommended minimum configuration for a successful implementation of DirXML.

- Server class with Pentium III – 400 MHz+
- RAM should be 2x DIB size plus 256 MB (1 GB minimum max of 2GB allocated to eDirectory)
- SYS: Volume configuration:
    - No compression
    - No sub-allocation
    - Threshold of 75% space used

Server assignment

The file server UTAH1 has been selected to host the DirXML engine. Its qualifications are as follows:

- Server class with Pentium III 1.4 GHz
- RAM is 2GB MB
- SYS: Volume configuration:
    - Compression is set to OFF
    - Block Sub-allocation is set to OFF
    - Disk space used is 140GB

UTSTATE eDirectory Requirements for DirXML

Design Specifications

The design of the UTSTATE eDirectory tree is for file and print resources spread across the State of Utah. For DirXML to synchronize to UMD, the DirXML engine must be located on a server (or servers) where replicas of all partitions exist of objects that need to be synchronized. Since users are the only objects that will need to be synchronized, only user replicas need to be placed on a single server. The directory version must also be a minimum of 8.6.2.

UTSTATE Tree Structure

Table to be inserted later

Figure 3: UTSTATE Tree Structure

Management Accounts

Beyond the normal administrative accounts DirXML should have an account to be equivalent to so as to track modifications made by the driver processes.

Account Name	Purpose	Rights Assigned
Admin	Administer the tree	Full administrator privileges (Entry Rights SBCRD) to [Root]
UMD User	Account used by DirXML for directory management in UTSTATE	Full administrator privileges (Entry Rights SBCRD) to [Root]

Hardware Configuration

Server assignment

The file server UTSTDPDS4 has been selected to host the DirXML engine. Its qualifications are as follows:

- Server class with Pentium III 1.2 GHz
- RAM is 2 GB
- SYS: Volume configuration:
    - Compression is set to off
    - Block Sub-allocation is set to off

- Disk space capacity is 70GB

UTCOURTS eDirectory Requirements for DirXML

Design Specifications

The design of the UTCOURTS eDirectory tree is for file and print resources spread across the State of Utah. For DirXML to synchronize to UMD, the DirXML engine must be located on a server (or servers) where replicas of all partitions exist of objects that need to be synchronized. Since users are the only objects that will need to be synchronized, only user replicas need to be placed on a single server. The directory version must also be a minimum of 8.6.2.

UTCOURTS Tree Structure

Table to be inserted later

Figure 4: UTCOURTS Tree Structure

Management Accounts

Beyond the normal administrative accounts DirXML should have an account to be equivalent to so as to track modifications made by the driver processes.

Account Name	Purpose	Rights Assigned
Admin	Administer the tree	Full administrator privileges (Entry Rights SBCRD) to [Root]
UMD User	Account used by DirXML for directory management in UTCOURTS	Full administrator privileges (Entry Rights SBCRD) to [Root]

Hardware Configuration

Server assignment

The file server ??? has been selected to host the DirXML engine. Its qualifications are as follows:

- Server class with Pentium ??? MHz
- RAM is ??? GB
- SYS: Volume configuration:
    - Compression is set to ???
    - Block Sub-allocation is set to ???

- Disk space used is 13%

UTDOT eDirectory Requirements for DirXML

Design Specifications

The design of the UDOT eDirectory tree is for file and print resources spread across the State of Utah. For DirXML to synchronize to UMD, the DirXML engine must be located on a server (or servers) where replicas of all partitions exist of objects that need to be synchronized. Since users are the only objects that will need to be synchronized, only user replicas need to be placed on a single server. The directory version must also be a minimum of 8.6.2.

UDOT Tree Structure

Table to be inserted later

Figure 5: UDOT Tree Structure

Management Accounts

Beyond the normal administrative accounts DirXML should have an account to be equivalent to so as to track modifications made by the driver processes.

Account Name	Purpose	Rights Assigned
Admin	Administer the tree	Full administrator privileges (Entry Rights SBCRD) to [Root]
UMD User	Account used by DirXML for directory management in UDOT	Full administrator privileges (Entry Rights SBCRD) to [Root]

Hardware Configuration

Server assignment

The file server ??? has been selected to host the DirXML engine. Its qualifications are as follows:

- Server class with Pentium ??? MHz
- RAM is ??? GB
- SYS: Volume configuration:
    - Compression is set to ???
    - Block Sub-allocation is set to ???

- Disk space used is 13%

USOR eDirectory Requirements for DirXML

Design Specifications

The design of the USOR eDirectory tree is for file and print resources spread across the State of Utah. For DirXML to synchronize to UMD, the DirXML engine must be located on a server (or servers) where replicas of all partitions exist of objects that need to be synchronized. Since users are the only objects that will need to be synchronized, only user replicas need to be placed on a single server. The directory version must also be a minimum of 8.6.2.

USOR Tree Structure

Figure 6: USOR Tree Structure

Management Accounts

Beyond the normal administrative accounts DirXML should have an account to be equivalent to so as to track modifications made by the driver processes.

Account Name	Purpose	Rights Assigned
Admin	Administer the tree	Full administrator privileges (Entry Rights SBCRD) to [Root]
UMD User	Account used by DirXML for directory management in USOR	Full administrator privileges (Entry Rights SBCRD) to [Root]

Hardware Configuration

Server assignment

The file server ??? has been selected to host the DirXML engine. Its qualifications are as follows:

- Server class with Pentium ??? MHz
- RAM is ??? GB
- SYS: Volume configuration:
    - Compression is set to ???
    - Block Sub-allocation is set to ???
    - Disk space used is 13%

HR – UMD (JDBC DirXML Driver)

DirXML Specifications

This JDBC DirXML Driver is responsible for creating and modifying user objects in the UMD tree based on employee events in the HR System. The HR System is based on Sybase and Novell Professional Services will be using jConnect 4.2 to communicate to an event log table that will monitor changes to a table that HR is populating.

Data Flow

The following diagram depicts the high-level data flow of the JDBC DirXML driver that is being created for the state of Utah.

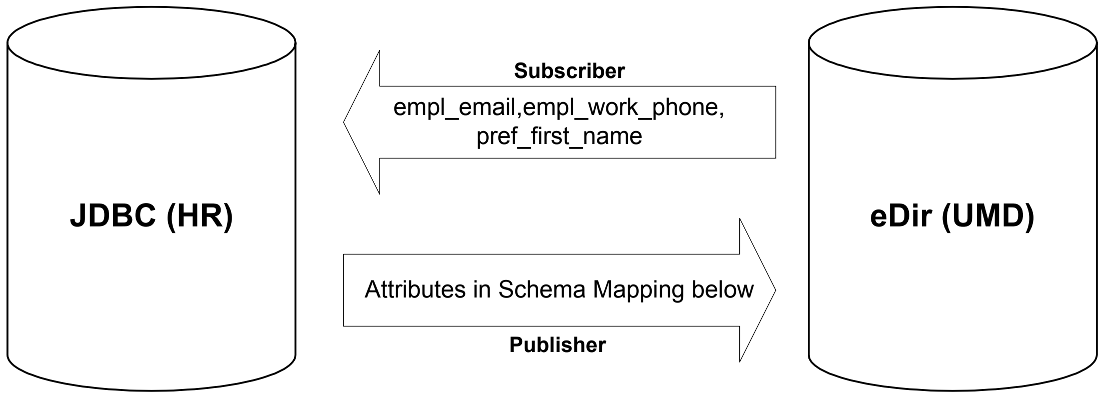
JDBC (HR)
Subscriber
empl_email,empl_work_phone, pref_first_name
Attributes in Schema Mapping below
Publisher
eDir (UMD)

Figure 7: Overall Data Flow (HR – UMD JDBC)

Driver Configuration Parameters

The communication between the DirXML host server and the HR System will be done with jConnect 4.2. Since this server and the HR system will be in the data center, SSL is not required per Curtis Parker.

See Appendix C for a list of all the network communication requirements.

The driver will be named and located as follows:

UMD : CN= HR.CN=DirXML Driver Set. O=IS

The following DirXML Driver Configuration Parameters have been set for the JDBC DirXML driver:

UMD

Driver Module

Java:	com.novell.nds.dirxml.driver.jdbc.JDBCDriverShim

Authentication

Authentication ID:	umd
Application Context:	jdbc:sybase:Tds:168.177.202.122:4100/HRSL Production jdbc:sybase:Tds:168.177.202.123:4100/HRSL t|Test

Startup Option

Auto start

Driver Parameters

Driver Settings	
Database Driver Class Name:	com.sybase.jdbc.SybDriver
Synchronize Schema:	dbo
Reuse Statements?:	Yes
Use Manual Transactions?:	Yes
Use Single Connection?:	No
Default Transaction Level:	read committed
Retrieve minimal meta data:	No
Use qualified table names?:	No
Handle Statement Results?:	Yes
Connection Initialization String:	USE HRSL
Connection Tester class name:	com.novell.nds.dirxml.driver.jdbc.util.JDBCConnectionTester
Subscriber Settings	
Disable?:	No
Primary Key Generation:	emp("sp_empno(empno, fname)")
Key Generation Timing:	after
Check Update Counts?:	Yes
Publisher Settings	
Disable?:	No
Log Table Name:	eventlog
Polling Interval(seconds):	10
Reconnect Interval(seconds):	30
Optimize Updates?:	Yes
Delete From Log?:	Yes
Allow Loopback?:	Yes

Schema Mapping

Schema Mappings provide a list of classes and attributes that should be mapped to each other.

The following table shows the schema mapping between the HR Sybase table and eDirectory (UMD).

Appendix A contains the customized schema within the UMD tree.

Table 1: Schema Mapping (HR – UMD JDBC)

UMD	HR	eDirectory Attribute Description	Authoritative Source	HR Filter	
				P	S
CN	n/a	Name of object (000+EIN)			
Department	agy_desc	Agency Description	HR	X	
Full Name	n/a	Full Name (created from Given Name and Surname)			
Generational Qualifier	empl_name_suffix	Suffix	HR	X	
Given Name	pref_first_name	Preferred First Name	HR	X	X
Initials	empl_second_name	Preferred Middle Initial	HR	X	
Internet EMail Address	empl_email	Internet Email generated from email system			X
manager	manager_ein	Manager’s DN Data Transform	HR	X	
Physical Delivery Office Name	work_addr_city	City	HR	X	
Postal Code	work_addr_zip	Zip Code	HR	X	
Postal Office Box	work_addr_line2	PO Box	HR	X	
Private Key	n/a	Password (Private key)			
Public Key	n/a	Password (Public Key)			
S	work_addr_state	State	HR	X	
SA	work_addr_line1	Street Address	HR	X	
Surname	empl_last_name	Last Name	HR	X	
Telephone Number	empl_work_phone	Business Telephone Number	HR	X	X
Title	jobb_title	Title	HR	X	
UTIW-AgencyNumber	empl_agy	Agency Number	HR	X	
UTIW-BldgID	work_building_id	Work Building ID	HR	X	
UTIW-DistList	dist_list	Distribution List for notification on user events	HR	X	
UTIW-EmpEmail	n/a	Suggested Employee Email account Data Transform	HR		
UTIW-InetMailDomain	email_inet_domain	Which Internet Domain Name	HR	X	
UTIW-InsurEligible	insurance_eligible	Insurance Eligible	HR	X	
UTIW-Invisible	protected_record_flag	Protected User in HR	HR	X	
UTIW-LeaveEligible	leave_eligible	Leave Eligible	HR	X	
UTIW-LowOrg	empl_org	Low Org	HR	X	
UTIW-NWAccess	empl_nw_access	Determines if Resource Tree account should be enabled or disabled.	HR	X	
UTIW-Placement	placement	initial placement to create user in Resource Tree	HR	X	
UTIW-PO	email_po	Which PO in Email	HR	X	
UTIW-PODomain	email_po_domain	Which Domain in the Email system	HR	X	
UTIW-PrimaryResTree	res_tree	Identifies the Primary Resource Tree	HR	X	
UTIW-PublicEmail	empl_public_email	Users Public Email address	HR	X	X

UMD	HR	eDirectory Attribute Description	Authoritative Source	HR Filter	
				P	S
UTIW-Status	empl_status	Employee Status	HR	X	
UTIW-UserTemplate	template	Which Template to build user in the resource tree	HR	X	
workforceID	employee_id	Employee Number (EIN)	HR	X	

Object Mapping and Placement

Object Mapping

The User objects will be associated by the value of the workforceID attributes within HR (matching rule). No matching rule will be needed going to HR as HR is the authoritative source of workforceIDs

Table 2: Object Mapping (HR-UMD JDBC)

HR to UMD	UMD to HR
User: workforceID	n/a

Create Rules

Table 3: Create Rules (HR-UMD JDBC)

HR to UMD	UMD to HR
Required attributes: workforceID, Given Name, Surname, UTIW-InetEmailDomain and UTIW-Status=A, N or I, UTIW-AgencyNumber, UTIW-LowOrg	n/a

Placement Rules

User objects from HR will be placed in the UMD tree based on the UTIW-Invisible attribute. Objects will not be placed in HR from UMD

Table 4: Placement Rules (HR-UMD JDBC)

HR	UMD
UTIW-Invisible = N	OU=Emp.OU=Gov.O=UT
UTIW-Invisible = Y	OU=Protect.OU=Gov.O=UT

Data Transformations

For this engagement, the following table summarizes all data transformations that will be performed by this driver:

Table 5: Data Transformations (HR-UMD JDBC)

UMD	HR	Transformation Description
CN	employee_id	Name of object (000+EIN)
Full Name	pref_first_name +	Full Name (created from Given Name and Surname)

UMD	HR	Transformation Description
	' '+empl_last_name	
Initials	empl_second_name	empl_second_name will be truncated to 8 Chars
manager	manager_ein	Manager's DN will be generated from a query based on the HR manager's workforceID for an associated user or left blank
Private Key	n/a	Password (Private key) Randomly Generated
Pubic Key	n/a	Password (Public key) Randomly Generated
Telephone Number	empl_work_phone	Formatted as XXX XXX-XXXX for UMD
UTIW-EmpEmail	pref_first_name, empl_second_name, empl_last_name, email_inet_domain	A unique internet email will be generated based on the following order of email prefix to determine a unique email name (including the UTIW-InetEmailDomain). It will be unique across UTIW-EmpEmail under the OU=Gov.O=UT container For all but email.utcourts.gov (i.e utah.gov): First Char of Given Name + Surname First Char of Given Name + First Char of Initials + Surname Given Name + Surname Given Name + First Char of Initials + Surname First Char of Given Name + Surname + # For email.utcourts.gov If UTIW-Invisible is True: First Char of Given Name + Surname First Char of Given Name + Surname +# If UTIW-Invisible is False: Given Name + First Char of Surname Given Name + First Char of Surname +#

General Specifications for HR - Utah Master Directory (UMD)

General info: The State of Utah's HR system will populate UMD based on Hires, Terminations and Updates of employees.

The following overall design guidelines were followed:

- The HR Sybase system is the authoritative source of all employees
- When an employee is hired, and synchronized to UMD, they will receive a suggested email account
- When an employee is created in UMD they will be created by a template CN=UMDEmpTemplate.OU=Gov.O=UT
- An employee's OrgID will determine:
    - What Internet email domain name will be used to create an email name (i.e. @utah.gov, @email.utcourts.gov, …)
    - What email post office the user's mail will be stored
    - What domain the post office is in
    - What resource tree the user will be created in
        - Unique emp_agy have the same res_tree
    - Where the user will be placed in the resource tree
    - Which template object the user will be created with
    - Which Distribution List to send email to for certain events
- empl_org is not unique between agencies but unique within an agency
- When an employee is created in the Utah Master Directory (UMD) a random password will be assigned that must be changed on the next login of the user

- HR will insert a new record into dbo.umd_empl_master when a new employee is hired. This record will also pull the associated OrgID information from dbo.umd_resource (as documented above)
- There is an empl_status field (mapped to UTIW-Status) that will have the following statuses:
    - A Active
    - I In-Active

      (Disable user in downstream resource trees and not allow access to some employee web pages)
    - T Terminate

      (Disable/delete user in downstream resource trees)
    - N New Hire

      (Do not create user in downstream resource trees by setting empl_nw_access to false)
- The empl_status will change from N to A on the effective date of the hire
- The empl_nw_access will disable all accounts in connected trees. SiteMinder and other security tools should look to this attribute to determine if the account should be allowed access to certain applications
- There is a protected_record_flag,UTIW-Invisible, that when set will cause the employee to be created in a protected container.
- When an employee's OrgID changes an email will be sent to the previous and the new Distribution lists.
- An environmental variable will exist (v_EmailOnTransfers) that will prevent emails from being sent on changes of OrgID for the two periods of time that mass changes occur on OrgID. This variable must be set previous to HR making the mass change and once the changes have occurred, setting it back for further emails to occur.
- Changes to the umd_resource_master will only effect new users created after the change.
- The umd_empl_master fields of res_tree, placement, template, dist_list, email_inet_domain, email_po, email_po_domain will only occur on changes to the empl_agy and empl_org by reading the associated agy and org row. In other words, if the users empl_agy and/or empl_org change, the other resource attributes will be repopulated from the resource table.
- Changes to the empl_agy that do not cause a change to the res_tree but do cause a change to the email_inet_domain will create a email hold and send an email to the current dist_list.
- Changes to the empl_org will not cause a change to the res_tree.
- Changes to the empl_org that cause a change to the email_inet_domain will create a email hold and send an email to the current dist_list.

Event Transform – HR events and effect on UMD Tree

The following section describes both the typical behavior of an action type, the requirements, and what transformations to the default driver behavior will take place.

Add Object

The following list provides the details on the how and when an add object action will be triggered. The trigger will generally be the addition of a new object to the originating application. When this action occurs, by default, a new associated object will be created in the destination application.

- User objects will only be created in the UMD tree if the required attributes are provided by the HR system.
- Required attributes for User objects: workforceID, Given Name, Surname, UTIW-InetEmailDomain and UTIW-Status=A, N or I, UTIW-AgencyNumber, UTIW-LowOrg.
    - If a matching rule(s) match exactly one object in UMD then the data in HR will overwrite any data in the UMD object for which HR is authoritative over. An association between these two objects will be established. The add event will be changed into a modify event.
    - If the empl_public_email from HR matches a corresponding user's Internet EMail Address in UMD's OU=Pub.O=UT sub containers or OU=Temp.OU=Gov.O=UT container:
        - The Pub/Temp user will be updated to contain the HR information where HR is authoritative.
        - The existing Internet EMail Address will be copied to UTIW-PublicEmail.
        - A UTIW-EmpEmail will be created based on the Data transform from above.
        - The user object will be renamed to 000+workforceID.
        - The user object will be moved to the appropriate container under OU=Gov.O=UT based on the placement rules.
    - If multiple matches are found in UMD, then an email (**Mail 1**) will be sent to the CN=UMD-Admin.OU=Trees.O=IS Internet EMail Address and an error will be logged in DirXML.log
    - If a match cannot be made in UMD, then a new object will be created
        - A new user will be created with the attributes and data transforms described above.
        - A Random Password will be generated for the user.
        - The new user will be placed in the appropriate container as determined by the placement rules above.
        - The UTIW-%PrimaryResTree auxiliary class will be added based on the UTIW-PrimaryResTree attribute.

Modify Object

The following list provides the details on how and when a modify object action will be triggered. The trigger will generally be the modification of an existing object in the originating application. When this action occurs, by default, the corresponding object in the destination application will be changed.

- If the UTIW-Status attribute changes to a T
    - An object will be created in OU=Hold.OU=Gov.O=UT by the name of the Internet EMail Address. The UTIW-EmpEmail on this object will be set to the Internet EMail Address.
    - The Internet EMail Address will be set to the value of UTIW-PublicEmail on the original object.
        - If the UTIW-PublicEmail does not exist the UTIW-EmpEmail will lose the last portion of the address (i.e. @utah)
    - UTIW-Employee, UTIW-Invisible, UTIW-UTSTATE, UTIW-UTCOURTS, UTIW-UDOT, UTIW-USOR and UTIW-USOE auxiliary classes will be removed if on the user object.
    - Other HR related attributes will be cleared including Department, manager, Physical Delivery Office Name, Postal Code, Postal Office Box, S, SA, Telephone Number, Title, UTIW-PrimaryResTree and workforceID.
    - The User object will be moved to sub container OU=#.OU=Pub.O=UT where # is the user's last digit on their CN attribute.
- If UTIW-Invisible changes, the user object will be moved based on the placement rules above.
- If the UTIW-PrimaryResourceTree changes, the old UTIW-(restree) auxiliary class will be removed and the new UTIW-(restree) auxiliary class will be added.
- If the UTIW-InetMailDomain changes:
    - Based on a change to the UTIW-AgencyNumber but not the UTIW-Uxxx aux class or just the UTIW-LowOrg will cause:
        - An object will be created in OU=Hold.OU=Gov.O=UT by the name of the Internet EMail Address. The UTIW-EmpEmail on this object will be set to the Internet EMail Address.
        - A new UTIW-EmpEmail will be generated for the user
        - An email (**Mail 2**) will be sent to the UTIW-DistList informing them that this employee had an Internet Domain Mail account of %Internet EMAIL Name and a new one will be created for the new %UTIW-InetMailDomain.
- Other modifications will update the associated UMD attributes based on the schema mappings and data transforms above.

Delete Object

The following list provides the details on how and when a delete object action will be triggered. The trigger will generally be the removal of an existing object in the originating application. When this action occurs, by default, the corresponding object will be deleted.

- This event should not happen to an associated user in HR. If a "rogue" process does delete an associated user:
    - The association will be removed and
    - An email (**Mail 3**) will then be sent to the CN=UMD-Admin.OU=Trees.O=IS Internet EMail Address.

Move Object

The following list provides the details on how and when a move object action will be triggered. The trigger will generally be a location change of an existing object in the originating application.

- The HR system does not have move functionality.
- The JDBC driver will be configured to block this event.

Rename Object

The following list provides the details on how and when a rename object action will be triggered. The trigger will generally be a naming attribute change to an existing object in the originating application. When this action occurs, by default, the corresponding object will be renamed.

- The HR system does not have rename functionality.
- The JDBC driver will be configured to block this event.

Event Transform – UMD events and effect on HR

The following section describes the typical behavior of an action type, the requirements and what transformations to the default driver behavior will take place.

General Specifications

- No objects created in UMD will create rows in the HR system
- Only updates to associated objects in UMD for the following fields will cause updates in the HR system:
    - Given Name
    - Internet EMail Address
    - UTIW-PublicEmail
    - Telephone Number

Add Object

The following list provides the details on the how and when an add object action will be triggered. The trigger will generally be the addition of a new object to the originating application. When this action occurs, by default, a new duplicate object will be created in the destination application.

- This event will not cause an effect in the HR System.
- The JDBC driver will be configured to block this event.

Note: This will allow:

- A contractor/vendor/volunteer process to create accounts in UMD without causing effects in the HR System
- The addition of public users to the Pub Container

Modify Object

The following list provides the details on how and when a modify object action will be triggered. The trigger will generally be the modification of an existing object in the originating application. When this action occurs, by default, the corresponding object in the destination application will be changed.

- Only updates to attributes in the Subscriber Filter will be allowed to update the HR System

Delete Object

The following list provides the details on how and when a delete object action will be triggered. The trigger will generally be the removal of an existing object in the originating application. When this action occurs, by default, the corresponding object will be deleted.

- This event will not cause an effect in the HR System.
- The JDBC driver will be configured to block this event.

Note: This will allow a contractor/vendor/volunteer process to delete accounts in UMD without causing effects in the HR System

Move Object

The following list provides the details on how and when a move object action will be triggered. The trigger will generally be a location change of an existing object in the originating application. When this action occurs, by default, the corresponding object will be deleted.

- This event will not cause an effect in the HR System.
- The JDBC driver will be configured to block this event.

Note: This will allow:

- A contractor/vendor/volunteer process to move accounts in UMD without causing effects in the HR System.
- The movement of user objects from the Pub to Emp container.

Rename Object

The following list provides the details on how and when a rename object action will be triggered. The trigger will generally be a naming attribute change to an existing object in the originating application. When this action occurs, by default, the corresponding object will be renamed.

- This event will not cause an effect in the HR System.
- The JDBC driver will be configured to block this event.

Note: This will allow:

- A contractor/vendor/volunteer process to rename accounts in UMD without causing effects in the HR System.
- The renaming of user objects when they are moved from the Pub to Emp container.

UMD – UTSTATE/UDOT/USOR eDirectory –eDirectory DirXML Driver

DirXML Specifications

This eDirectory to eDirectory DirXML Driver is responsible for creating and modifying user objects in the UTSTATE/UDOT/USOR (here on known as Uxxx) Tree based on user events in the UMD Tree.

Data Flow

The following diagram depicts the high-level data flow of the eDirectory-eDirectory driver that is being created for the state of Utah.

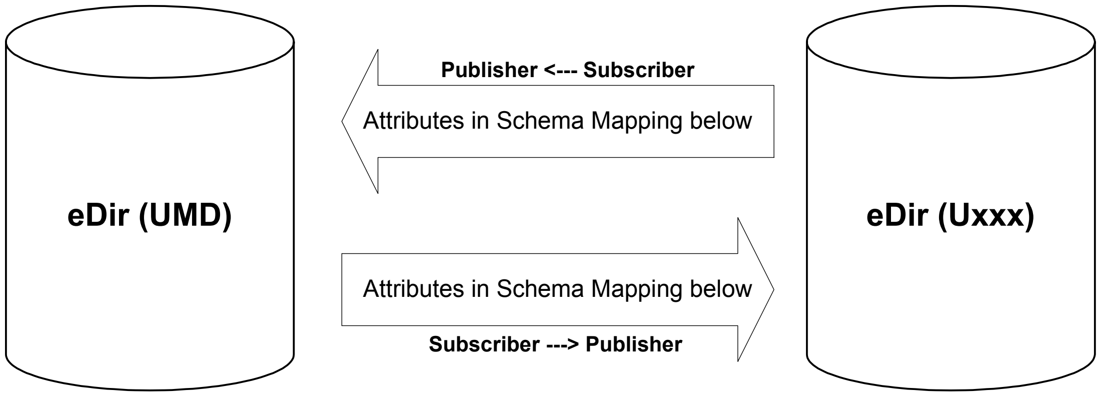
eDir (UMD)
Publisher <--- Subscriber
Attributes in Schema Mapping below
Attributes in Schema Mapping below
Subscriber ---> Publisher
eDir (Uxxx)

Figure 8: Overall Data Flow (eDirectory-eDirectory)

Driver Configuration Parameters

Each eDirectory driver host server will need to communicate with a corresponding eDirectory server holding a replica of the desired partitions to be synchronized.

The communication between the eDirectory driver host servers will be over an encrypted channel. This implementation will use a single KMO per tree (DirXML Certificate for both trees) with UMD being the master tree   .

See Appendix C for a list of all the network communication requirements.

The driver will be named and located as follows:

UMD: CN= Uxxx.CN=DirXML Driver Set. O=IS

Uxxx: CN= UMD.CN=DirXML Driver Set.O= IS

The following DirXML Driver Configuration Parameters have been set for the eDirectory-eDirectory drivers:

UMD

Driver Module

Java:	com.novell.eDirectory.dirxml.driver.eDirectory.DriverShimImpl

Authentication

Authentication ID:	DirXML Certificate Uxxx
Application Context:	IPADDR:2000 (UTSTATE)
	IPADDR:2001 (UDOT)
	IPADDR:2002 (USOR)

Startup Option

Auto start

Uxxx

Driver Module

Java:	com.novell.eDirectory.dirxml.driver.eDirectory.DriverShimImpl

Authentication

Authentication ID:	DirXML Certificate Uxxx
Application Context:	UTAH1:2000 (UTSTATE)
	UTAH1:2001 (UDOT)
	UTAH1:2002 (USOR)

Startup Option

Auto start

Schema Mapping

Schema Mappings provide a list of classes and attributes are mapped to each other.

The following table shows the schema mapping between the eDirectory (UMD) and eDirectory (Uxxx). The UMD and Uxxx trees are both using User as their base object class. The attribute names used are in the eDirectory format.

Appendix A contains the customized schema within the UMD tree. Uxxx did not need a customized schema for this engagement.

Table 6: Schema Mapping (eDirectory-eDirectory)

UMD	Uxxx	eDirectory Attribute Description	Authoritative Source	UMD Filter		Uxxx Filter	
				P	S	P	S
<all classes>: GUID	<all classes>: GUID	Primary key in each Tree for Association			X		X
Description	Description	Description of User			X	X	

UMD	Uxxx	eDirectory Attribute Description	Authoritative Source	UMD Filter		Uxxx Filter	
				P	S	P	S
Facsimile Telephone Number	Facsimile Telephone Number	Fax Number		X	X	X	X
Full Name	Full Name	Full Name			X	X	
Generational Qualifier	Generational Qualifier	Suffix			X	X	
Given Name	Given Name	Preferred First Name		X	X	X	X
Initials	Initials	Preferred Middle Initial	HR		X	X	
Internet EMail Address	Internet EMail Address	Internet Email generated from email system	Email System	X			X
Login Disabled	Login Disabled	Login Disabled			X	X	
mailstop	mailstop	MailStop		X	X	X	X
mobile	mobile	Cell Phone		X	X	X	X
pager	pager	Pager number		X	X	X	X
Password Required	Password Required	Require user to have password			X	X	
Physical Delivery Office Name	Physical Delivery Office Name	City	HR		X	X	
Postal Code	Postal Code	Zip Code	HR		X	X	
Postal Office Box	Postal Office Box	PO Box	HR		X	X	
Private Key	Private Key	Password (Private key)		X	X	X	X
Public Key	Public Key	Password (Public Key)		X	X	X	X
S	S	State	HR		X	X	
SA	SA	Street Address	HR		X	X	
Surname	Surname	Last Name	HR		X	X	
Telephone Number	Telephone Number	Business Telephone Number		X	X	X	X
Title	Title	Title (From HR the title will be sent to Uxxx on creation only. Modifications on title in UMD will be blocked.		X	X	X	X
UTIW-AgencyNumber	n/a	Agency Number	HR		X	X	
UTIW-EmpEmail	CN (add only)	Email Prefix will be the CN in Uxxx			X	X	
UTIW-InetMailDomain	n/a	Which Domain in Email	HR		X	X	
UTIW-Invisible	n/a	Protected User in HR	HR		X	X	
UTIW-LowOrg	n/a	Low Org	HR		X	X	
UTIW-NWAccess	n/a	Determines if Resource Tree account is enabled for login.	HR		X		
UTIW-Placement	n/a	initial placement to create user in Resource Tree	HR		X	X	
UTIW-udotCN	n/a	CN of object in UDOT	UDOT	X			
UTIW-udotPlacement	n/a	Placement of object in UDOT	UDOT	X			
UTIW-UserTemplate	n/a	Which Template to build user	HR		X	X	

UMD	Uxxx	eDirectory Attribute Description	Authoritative Source	UMD Filter		Uxxx Filter	
				P	S	P	S
UTIW-usorCN	n/a	CN of object in USOR	USOR	X			
UTIW-usorPlacement	n/a	Placement of object in USOR	USOR	X			
UTIW-utcourtsCN	n/a	CN of object in UTCOURTS	UTCOURTS	X			
UTIW-utcourtsPlacement	n/a	Placement of object in UTCOURTS	UTCOURTS	X			
UTIW-utstateCN	n/a	CN of object in UTSTATE	UTSTATE	X			
UTIW-utstatePlacement	n/a	Placement of object in UTSTATE	UTSTATE	X			
workforceID	workforceID	Employee Number (EIN)	HR		X	X	

Object Mapping and Placement

Object Mapping

The User objects will be associated by the value of the workforceID attributes within both trees (matching rule). Going from UMD to Uxxx is a just match on workforceID while going from Uxxx back to UMD will also require a match on Surname and the appropriate auxiliary class already defined in UMD (i.e. objectclass=UTIW-Uxxx .)

Table 7: Object Mapping (eDirectory-eDirectory)

UMD to Uxxx	Uxxx to UMD
User: workforceID	User: workforceID, Surname and Aux Class match

Create Rules

The following attributes are required:

Table 8: Create Rule(s) (eDirectory-eDirectory)

UMD to Uxxx	Uxxx to UMD
UTIW-EmpEmail, Given Name, Surname, Public/Private Key is not blank (password is required), Object Class=UTIW-Uxxx, UTIW-Placement (must also be valid) and UTIW-PrimaryResTree=Uxxx.	Title=non-state

Placement Rules

User objects in the UMD tree will be placed in the UTSTATE based on the UTIW-Placement attribute. Objects from UTSTATE will only match to existing UMD users so no placement will be required

Table 9: Placement Rules (eDirectory-eDirectory)

UMD to Uxxx	Uxxx to UMD

UMD to Uxxx	Uxxx to UMD
Users under OU=Gov.O=UT: based on UTIW-Placement	Title=non-state will place in OU=Temp.OU=Gov.O=UT
Users elsewhere: not placed	Will only be matched to existing user based on a match of workforceID and Surname

Data Transformations

For this engagement, the following table summarizes all data transformations that will be performed by this driver:

Table 10: Data Transformations (eDirectory-eDirectory)

UMD	Uxxx	Transformation Description
mobile	mobile	UMD will store the phone number in the format of XXX XXX-XXXX
pager	pager	UMD will store the phone number in the format of XXX XXX-XXXX
Telephone Number	Telephone Number	UMD will store the phone number in the format of XXX XXX-XXXX
UTIW-EmpEmail	CN	The CN will be the email prefix from UTIW-EmpEmail during add operations only

General Specifications for UMD to Uxxx (UTSTATE/UDOT/USOR)

General info: The State of Utah's HR system will populate UMD based on hires, terminations and updates.

All references to %*attributename* will have the value of the attribute of the user object.

The following overall design guidelines were followed:

- Changes in UMD will be done through approved interfaces following the guidelines specified below.
- Agencies will never change Primary Resource Trees without changing emp_agy (UTIW-AgencyNumber)
- UTIW-AgencyNumber and UTIW-LowOrg are numeric fields and will not contain text.
- Mass updates of OrgIDs will cause significant emails unless the v_EmailOnTransfers is set to False in the driver(s) XSLT stylesheets.

  Note: failure to return this value to true will result in no new UTIW-LowOrg transfer notifications when the placement does not change.

Event Transform – UMD events and effect on Uxxx Tree

The following section describes both the typical behavior of an action type, the requirements, and what transformations to the default driver behavior will take place.

From UMD, the following events will be passed to Uxxx.

Add Object

The following list provides the details on the how and when an add object action will be triggered. The trigger will generally be the addition of a new object to the originating application. When this action occurs, by default, a new associated object will be created in the destination application.

- This driver will only monitor add events under OU=Gov.O=UT.
- When UTIW-Uxxx auxiliary class exists,
    - If UTIW-NWAccess is false, the event will be blocked, otherwise
        - If the matching rule(s) match more than one object in Uxxx, an error will be logged in dirxml.log and an email (**Mail 1**)will be sent to the CN=Uxxx-Admin.OU=Trees.O=IS Internet EMail Address.
    - If the matching rule(s) match exactly one object in Uxxx, the add event will be changed into a modify event and processed as described in the Modify Object section.
    - If a match cannot be made in Uxxx then a new object will be created in Uxxx:
        - User objects will only be created in the Uxxx tree if the required attributes exist in the user object and the %UTIW-Placement exists in Uxxx
            - Required attributes for User objects are: UTIW-EmpEmail, Surname, Public/Private Key is not blank (password is required), Object Class=UTIW-Uxxx, UTIW-Placement (must also be valid) and UTIW-PrimaryResTree=Uxxx.
        - If the UTIW-Placement does not exist, an error will be logged in DirXML.log
        - The Uxxx CN will be set to the UTIW-EmpEmail prefix provided that the CN is not already taken in Uxxx in the %UTIW-Placement. If it is, the CN will be regenerated by querying the UMD tree to generate a unique CN based on the email prefix rules from HR. The UTIW-EmpEmail will then be updated in UMD based on the new CN.
        - The User object will be created with a template object (%UTIW-User-Template if present) to set default login values, group memberships, other attributes and create a home directory.
        - The User object will be created with a Login Time of April 1, 1990 to comply with auditing (accounts with login times 60 days previous do not count as a licensed account.)
        - If a UTIW-PrimaryResTree remove value exists, this was a transfer event and a random password will not be generated.
            - An email (**Mail 2)** will be sent to the manager's email address and to the UTIW-DistList
        - Otherwise, this was a new hire and a random password will be generated.

- An email (**Mail 3)** will be sent to the manager’s email address and to the UTIW-DistList.

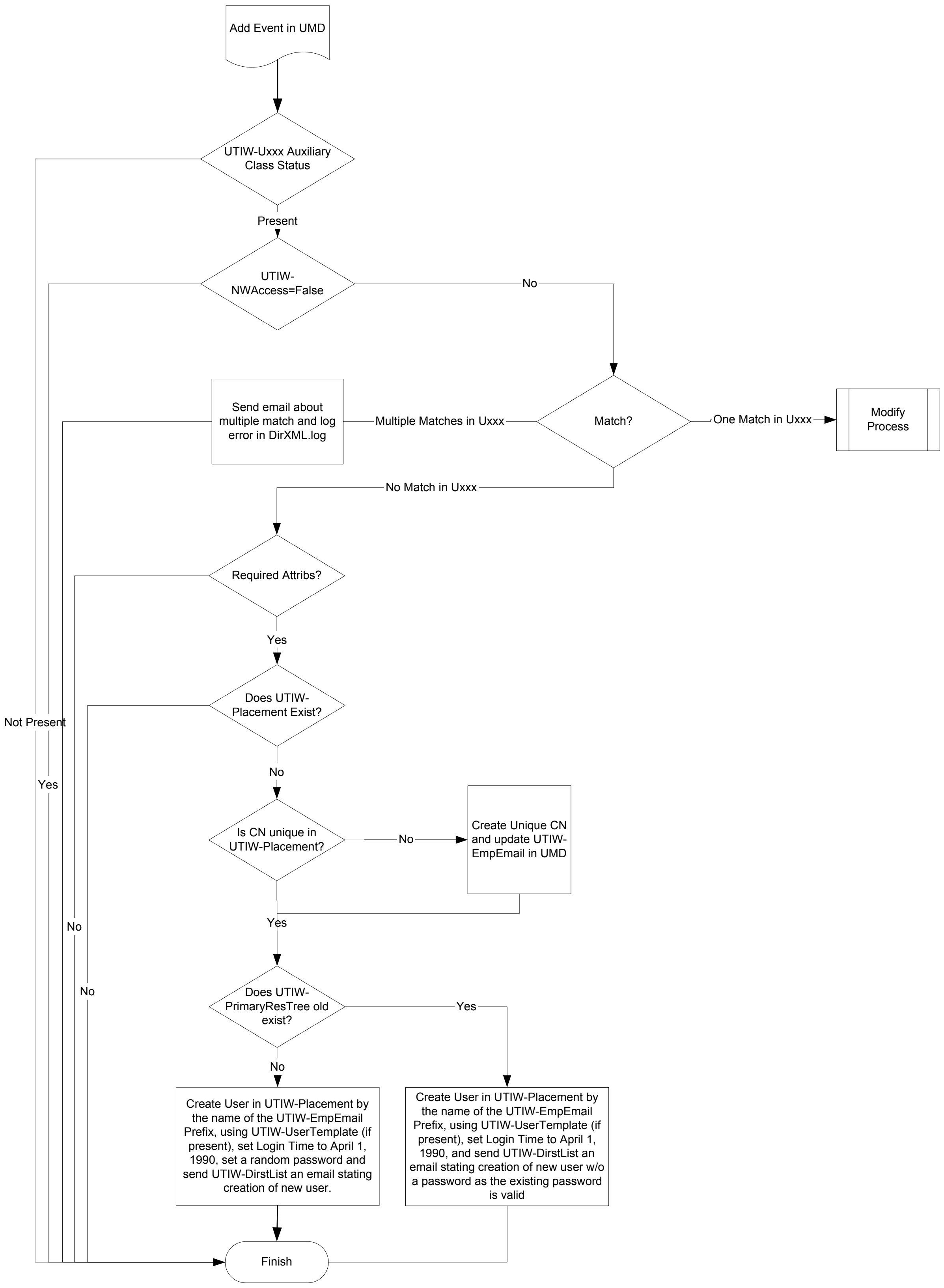
Add Event in UMD
UTIW-Uxxx Auxiliary Class Status
Present
Not Present
UTIW-NWAccess=False
No
Yes
Match?
One Match in Uxxx
Modify Process
Multiple Matches in Uxxx
Send email about multiple match and log error in DirXML.log
No Match in Uxxx
Required Attribs?
Yes
No
Does UTIW-Placement Exist?
No
No
Is CN unique in UTIW-Placement?
No
Create Unique CN and update UTIW-EmpEmail in UMD
Yes
Does UTIW-PrimaryResTree old exist?
Yes
No
Create User in UTIW-Placement by the name of the UTIW-EmpEmail Prefix, using UTIW-UserTemplate (if present), set Login Time to April 1, 1990, set a random password and send UTIW-DirstList an email stating creation of new user.
Create User in UTIW-Placement by the name of the UTIW-EmpEmail Prefix, using UTIW-UserTemplate (if present), set Login Time to April 1, 1990, and send UTIW-DirstList an email stating creation of new user w/o a password as the existing password is valid
Finish

Figure 9: UMD Add to Uxxx Event

Modify Object

The following list provides the details on how and when a modify object action will be triggered. The trigger will generally be the modification of an existing object in the originating application. When this action occurs, by default, the corresponding object in the destination application will be changed.

- If the UTIW-Uxxx object class and association is removed (UTIW-Status = T) from an existing object, the account in Uxxx will be disabled and an email (**Mail 4**) will be sent to the UTIW-DistList
- If UTIW-NWAccess= False then Disable associated user and send email (**Mail 5**) to UTIW-DistList that HR has disabled the user.
- If the UTIW-Uxxx object class is added to an existing object and the UTIW-PrimaryResTree is different than Uxxx, the event will be blocked. Otherwise:
    - If the UTIW-AgencyNumber changes check the UTIW-PrimaryResTree.
        - If the UTIW-PrimaryResTree is different
            - Follow add process for new UTIW-PrimaryResTree
            - The current associated object will have
                - The workforceID cleared
                - The association removed
                - Renamed to old-%CN
                - A user created in UMD with the name of the existing Internet EMail Address in OU=Hold.OU=Gov.O=UT whose UTIW-EmpEmail is the same as the Internet EMail Address.
                - An email (**Mail 6** mail 1 in diag) sent to the previous and current UTIW-DistList.
        - Otherwise, if the UTIW-Placement changes send an email to the previous and current UTIW-DistList (**Mail 7** mail 2 in diag):
    - If the UTIW-LowOrg and UTIW-Placement both change, an email will be sent to the previous and current UTIW-DistList about the change (**Mail 8** mail 3 in diag)
    - If the UTIW-LowOrg changes and the UTIW-Placement does not change, an email (**Mail 9** mail 4 in diag) will be sent to the previous and current UTIW-DistList about the change unless the v_EmailOnTransfers is False.
- Modifications to User attributes that are in the Uxxx Subscriber filter and the UMD Publisher filter (see Schema above) will cause modifications to the corresponding attributes on the associated object in Uxxx.
- If no association exists or an invalid association exists the Modify event will turn into an Add event following the rules stated above.

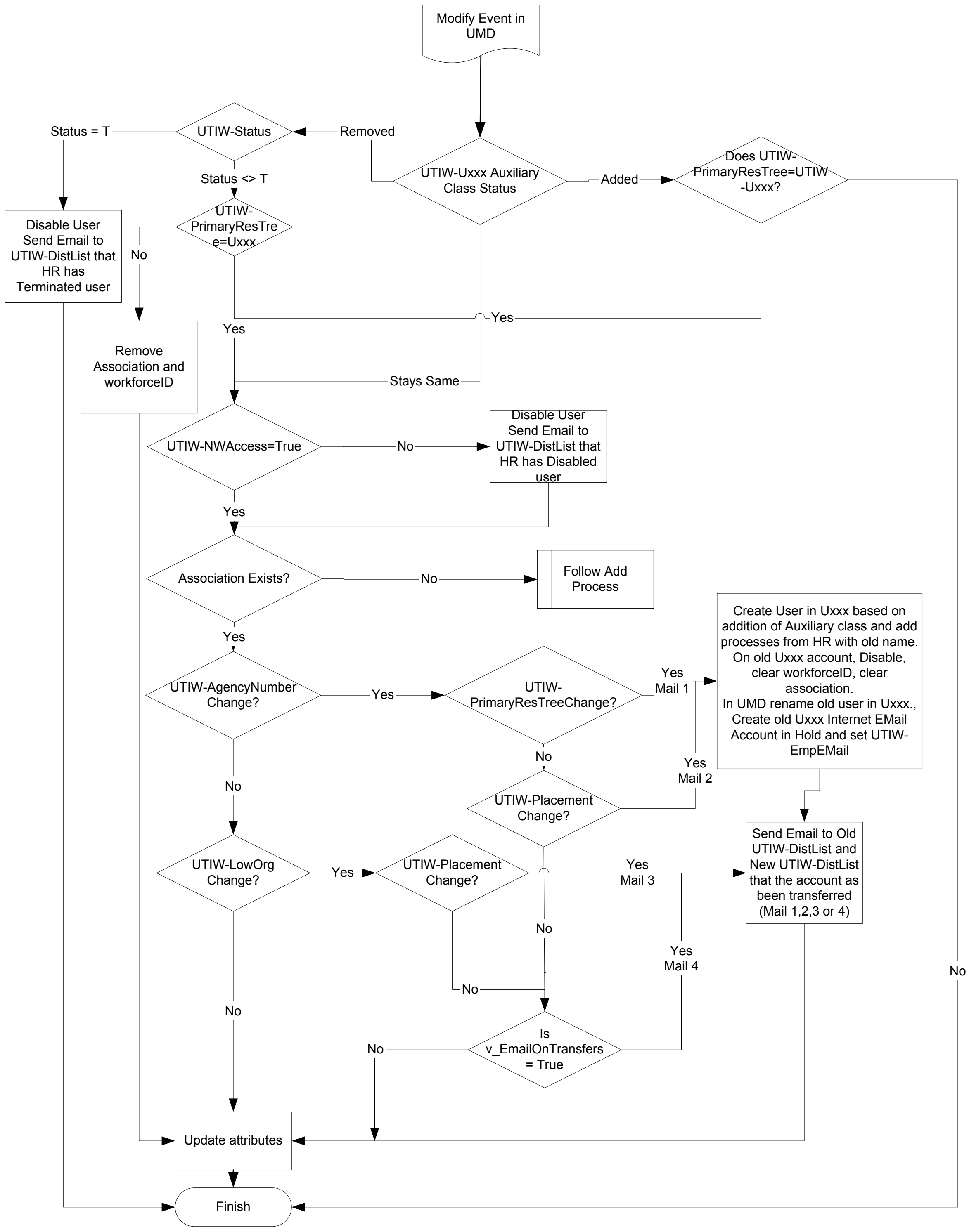
Modify Event in UMD
UTIW-Uxxx Auxiliary Class Status
Removed
UTIW-Status
Status = T
Disable User Send Email to UTIW-DistList that HR has Terminated user
Status <> T
UTIW-PrimaryResTree=Uxxx
No
Remove Association and workforceID
Added
Does UTIW-PrimaryResTree=UTIW-Uxxx?
Yes
Yes
Stays Same
UTIW-NWAccess=True
No
Disable User Send Email to UTIW-DistList that HR has Disabled user
Yes
Association Exists?
No
Follow Add Process
Yes
UTIW-AgencyNumber Change?
Yes
UTIW-PrimaryResTreeChange?
Yes Mail 1
Create User in Uxxx based on addition of Auxiliary class and add processes from HR with old name. On old Uxxx account, Disable, clear workforceID, clear association. In UMD rename old user in Uxxx., Create old Uxxx Internet EMail Account in Hold and set UTIW-EmpEMail
No
UTIW-Placement Change?
Yes Mail 2
No
UTIW-LowOrg Change?
Yes
UTIW-Placement Change?
Yes Mail 3
Send Email to Old UTIW-DistList and New UTIW-DistList that the account as been transferred (Mail 1,2,3 or 4)
No
Yes Mail 4
No
No
No
Is v_EmailOnTransfers = True
No
No
Update attributes
Finish

Figure 10: UMD Modify to Uxxx Event

Delete Object

The following list provides the details on how and when a delete object action will be triggered. The trigger will generally be the removal of an existing object in the originating application. When this action occurs, by default, the corresponding object will be deleted.

- Deleted user objects should only occur from the Public Container
- This event should not happen to a user in OU=Gov.O=UT that has an association in Uxxx. If a "rogue" process does delete an associated user:
    - The event will be modified into a disable login of the Uxxx user and remove association.
    - An email (**Mail 10**) will then be sent to the CN= Uxxx-Admin.OU=Trees.O=IS Internet EMail Address.

Move Object

The following list provides the details on how and when a move object action will be triggered. The trigger will generally be a location change of an existing object in the originating application.

- If an associated user is moved within the OU=Gov.O=UT sub containers the move message will be blocked.
- If an associated user is moved outside of the OU=Gov.O=UT container a "rogue" process has occurred.
    - The event will be modified into a disable login of the Uxxx user.
    - An email (**Mail 11**) will then be sent to the CN=Uxxx-Admin.OU=Trees.O=IS Internet EMail Address.

Rename Object

The following list provides the details on how and when a rename object action will be triggered. The trigger will generally be a naming attribute change to an existing object in the originating application.When this action occurs, by default, the corresponding object will be renamed.

- Any Rename event in UMD will have no effect in Uxxx.
- The event will be blocked.

Event Transform – Uxxx events and effect on UMD

The following section describes the typical behavior of an action type, the requirements and what transformations to the default driver behavior will take place.

General Specifications

- New User objects should only be created via UMD. A user should only be created in Uxxx if HR has not processed the user due to extraordinary events. Once HR has created the user, the manager and LAN-Admins (DistList) will be receiving an email concerning an account that has been created via the automated processes described above. A LAN-Admin should either:

  - Write down the workforceID of the newly created user account. Delete the newly created user. Add the workforceID to the previously created account if the Surname is the legal name of the user. Otherwise the Surname must be changed to the legal name of the user prior to adding the workforceID.
  - Or, Delete the previously created account. Rename the newly created account if necessary.

Add Object

The following list provides the details on the how and when an add object action will be triggered. The trigger will generally be the addition of a new object to the originating application. When this action occurs, by default, a new duplicate object will be created in the destination application.

- Contractor/Vendor/Volunteers (CVV) who need an email account in the current environment (before GW integration is done) must have their title set to 'non-state'. This will allow the driver to create a user in OU=Temp.OU=Gov.O=UT and keep the Internet EMail Address (and UTIW-EmpEmail) as used so as to allow the JDBC driver to correctly pick a unique suggested email account. The CVV Process needs to be followed instead.
- With the exception of the above, User objects will not be created in the UMD tree from this event but may be associated to an existing object in UMD as long as the follow occurs: Object section.
  - If the workforceID matches a user in OU=Gov.O=UT but the Surname does not, the workforceID will be cleared in Uxxx.
  - If the workforceID and Surname match a user in OU=Gov.O=UT but an association to an existing user in the Uxxx tree exists, the workforceID will be cleared in Uxxx. An email (**Mail 12**) will also be sent to both CN=Uxxx-Admin.OU=Trees.O=IS and CN=UMD-Admin.OU=Trees.O=IS Internet EMail Addresses stating that an existing user exists in the Uxxx tree with an association.
  - If the workforceID and Surname match a user in OU=Gov.O=UT and an association does not exist to an existing user in Uxxx, the auxiliary class for UTIW-Uxxx must exist in order to make an association between the two objects otherwise the workforceID will be cleared and an email (**Mail 13**)will also be sent to both CN=Uxxx-Admin.OU=Trees.O=IS and CN=UMD-Admin.OU=Trees.O=IS Internet EMail Addresses stating that an existing user exists in the Uxxx tree with an association this.
    - The attributes in UMD will take precedence over the attributes in Uxxx. (a merge of multi-value attributes will not be allowed)

Note: Business Process prefers the LAN-Admin contact HR to get the user created through the automated process rather than manually creating the user.

Modify Object

The following list provides the details on how and when a modify object action will be triggered. The trigger will generally be the modification of an existing object in the originating application. When this action occurs, by default, the corresponding object in the destination application will be changed.

- User objects will not be created in the UMD tree from this event but may be associated to an existing object in UMD as long as the follow occurs: Object section.
    - If the workforceID matches a user in OU=Gov.O=UT but the Surname does not, the workforceID will be cleared in Uxxx.
    - If the workforceID and Surname match a user in OU=Gov.O=UT but an association to an existing user in Uxxx exists, the workforceID will be cleared in Uxxx. An email (**Mail 12**) will also be sent to both CN=Uxxx-Admin.OU=Trees.O=IS and CN=UMD-Admin.OU=Trees.O=IS Internet EMail Addresses stating that an existing user exists in the Uxxx tree with an association.
    - If the workforceID and Surname match a user in OU=Gov.O=UT and an association does not exist to an existing user in Uxxx, the auxiliary class for UTIW-Uxxx must exist in order to make an association between the two objects otherwise the workforceID will be cleared and an email (**Mail 13**)will also be sent to both CN=Uxxx-Admin.OU=Trees.O=IS and CN=UMD-Admin.OU=Trees.O=IS Internet EMail Addresses stating that an existing user exists in the Uxxx tree with an association this.
        - The attributes in UMD will take precedence over the attributes in Uxxx. (a merge of multi-value attributes will not be allowed)
- If the Internet EMail Address is modified in Uxxx the UTIW-EmpEmail will also be set to the Internet EMail Address
- If the object already has an association to UMD the modifications will be synchronized according to the filters.

Delete Object

The following list provides the details on how and when a delete object action will be triggered. The trigger will generally be the removal of an existing object in the originating application. When this action occurs, by default, the corresponding object will be deleted.

- Delete events in the Uxxx on associated users will :
    - The user in UMD will have:
        - The association removed
        - If the UTIW-PrimaryResTree is Uxxx:
            - A User will be created in OU=Hold.OU=Gov.O=UT by the name of the Internet EMail Address and the

        UTIW-EmpEMail will be set to the Internet EMail Address.

        - The Internet EMail Address deleted on the UMD user.

    - If the UTIW-PrimaryResTree is not Uxxx:
        - The UTIW-Uxxx auxiliary class will be deleted from UMD

- If the user in UMD is not in a OU=Gov.O=UT sub container, the UTIW-Uxxx auxiliary class will be removed as will the association.

Move Object

The following list provides the details on how and when a move object action will be triggered. The trigger will generally be a location change of an existing object in the originating application. When this action occurs, by default, the corresponding object will be deleted.

- Move events in the Uxxx will be changed into a modify event to modify the UTIW-uxxxPlacement in UMD if the object is in OU=Gov.O=UMD

Rename Object

The following list provides the details on how and when a rename object action will be triggered. The trigger will generally be a naming attribute change to an existing object in the originating application. When this action occurs, by default, the corresponding object will be renamed.

- Rename events in the Uxxx will be changed into a modify event to modify the UTIW-uxxxCN in UMD if the object is in OU=Gov.O=UMD

Change Log

Date	Description	Requested By
05-23-2002	Initial Creation	Novell Team
06-04-02	Delivered First Draft to Darrus McBride	Novell Team
06-04-02	Changed Transfer of Agency UMD to Uxxx modify section to rename old and keep name on new account. Updated Mail 1 and Mail 2 to reflect this change.	NUI Group
06-16-2002	page 2 last paragraph verbiage changed page 6 LWoP added last paragraph page 11 Index on workforceID, UTIW-EmpEmail and UTIW-PubEmail page 15 UMD server specs added page 16 UTSTATE server specs added page 25 JDBC table updated with correct attribute names page 27 UMD users created from HR will be created with a template UMDEmpTemplate page 28 5 additional clarifications at end of general specifications page 29 Add specs verbiage clarified page 30 if modify causes object to go to Pub and UTIW-PublicEmail is null, drop the last of the UTIW-EmpEmail. page 30 clarification on changes to UTIW-InetDomainName page 36 updated Uxxx schema table to reflect proper schema page 37 Create Rules updated to include given name and also Title=non-state for contractors. page 38 Placement rules updated for non-state cvv in Uxxx page 38 General Assumption updated to reflect that Agency and LowOrg will contain only numbers page 40 update to email message on add scenario page 44 Updated Modify Graphic for scenario found during testing page 47 CVV add process for non-state Title added page 49 Delete in Uxxx updated for users concerning UTIW-PrimaryResTree.	Curtis Parker/Novell Team
06-17-02	Mail moved to external document for easier maintenance	David R. Lee

Appendix A - Schema Definition

OID	Name	Syntax	Description
Classes			
2.16.840.1.113719.2.221.6.1	UTIW-UTCOURTS	Auxiliary Class	Trigger Aux Class for Creating User in Resource Tree - UTCOURTS
2.16.840.1.113719.2.221.6.2	UTIW-UTSTATE	Auxiliary Class	Trigger Aux Class for Creating User in Resource Tree - UTSTATE
2.16.840.1.113719.2.221.6.3	UTIW-USOR	Auxiliary Class	Trigger Aux Class for Creating User in Resource Tree - UTSOR
2.16.840.1.113719.2.221.6.4	UTIW-UDOT	Auxiliary Class	Trigger Aux Class for Creating User in Resource Tree - UDOT
2.16.840.1.113719.2.221.6.5	UTIW-USOE	Auxiliary Class	Trigger Aux Class for Creating User in Resource Tree - USOE
2.16.840.1.113719.2.221.6.6	UTIW-Status	Auxiliary Class	Aux Class for Storing Users Primary Resource Tree email Notification Distribution List
2.16.840.1.113719.2.221.6.7	UTIW-Invisible	Auxiliary Class	Aux Class for Determining User's GroupWise Visibility and Placement in UMD Tree
2.16.840.1.113719.2.221.6.8	UTIW-Employee	Auxiliary Class	Aux Class for State of Utah Employees
2.16.840.1.113719.2.221.6.9	UTIW-Public	Auxiliary Class	Aux Class for State of Utah Public Users
2.16.840.1.113719.2.221.6.10	UTIW-CVV	Auxiliary Class	Aux Class for State of Utah Contract Users

OID	Name	Syntax	Description
Attributes			
2.16.840.1.113719.2.221.4.1	UTIW-utcourtsCN	CIS Single SI	DN of Users Current Placement in UTCOURTS Tree
2.16.840.1.113719.2.221.4.2	UTIW-utstateCN	CIS Single SI	DN of Users Current Placement in UTSTATE Tree
2.16.840.1.113719.2.221.4.3	UTIW-usorCN	CIS Single SI	DN of Users Current Placement in UTSOR Tree
2.16.840.1.113719.2.221.4.4	UTIW-udotCN	CIS Single SI	DN of Users Current Placement in UTDOT Tree
2.16.840.1.113719.2.221.4.5	UTIW-usoeCN	CIS Single SI	DN of Users Current Placement in USOE Tree
2.16.840.1.113719.2.221.4.6	UTIW-utcourtsPlacement	CIS Single SI	DN of Users Current Placement in UTCOURTS Tree
2.16.840.1.113719.2.221.4.7	UTIW-utstatePlacement	CIS Single SI	DN of Users Current Placement in UTSTATE Tree
2.16.840.1.113719.2.221.4.8	UTIW-usorPlacement	CIS Single SI	DN of Users Current Placement in UTSOR Tree
2.16.840.1.113719.2.221.4.9	UTIW-udotPlacement	CIS Single SI	DN of Users Current Placement in UTDOT Tree
2.16.840.1.113719.2.221.4.10	UTIW-usoePlacement	CIS Single SI	DN of Users Current Placement in USOE Tree

OID	Name	Syntax	Description
Attributes			
2.16.840.1.113719.2.221.4.11	UTIW-Status	CIS Single SI	Stores Employees Status A I N or T
2.16.840.1.113719.2.221.4.12	UTIW-DistList	CIS Single SI	Stores Users Primary Resource Tree email Notification Distribution List
2.16.840.1.113719.2.221.4.13	UTIW-PrimaryResTree	CIS Single SI	Intial Primary Tree
2.16.840.1.113719.2.221.4.14	UTIW-Invisible	Boolean SI	Determines Users GroupWise Visibility and Placement in UMD Tree
2.16.840.1.113719.2.221.4.15	UTIW-Placement	CIS Single SI	Intial Placement in Primary Tree
2.16.840.1.113719.2.221.4.16	UTIW-AgencyNumber	CIS Single SI	Users Agency Number
2.16.840.1.113719.2.221.4.17	UTIW-LowOrg	CIS Single SI	Users Organizational ID
2.16.840.1.113719.2.221.4.18	UTIW-UserTemplate	CIS Single SI	DN of Template used to create user in Primary Resource Tree
2.16.840.1.113719.2.221.4.19	UTIW-InetMailDomain	CIS Single SI	Internet Email Domain - example: @Utah.gov
2.16.840.1.113719.2.221.4.20	UTIW-PO	CIS Single SI	CN of Post Office to create users email Account
2.16.840.1.113719.2.221.4.21	UTIW-PODomain	CIS Single SI	CN of Domain that holds PO
2.16.840.1.113719.2.221.4.22	UTIW-NWAccess	Boolean SI	Determines if User is recreated in resource trees
2.16.840.1.113719.2.221.4.23	UTIW-PublicEmail	CIS Single SI	Used to assoc preexisting public id to a employee ID
2.16.840.1.113719.2.221.4.24	UTIW-EmpEmail	CIS Single SI	Used to hold employee email address in addition to Internet EMail Address, for email swap
2.16.840.1.113719.2.221.4.25	UTIW-InsurEligible	CIS Single SI	Used to determine if the employee is Insurance Eligible

UMD Class Definitions for UTIW-UTCOURTS		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-utcourtsCN
		UTIW-utcourtsPlacement

UMD Class Definitions for UTIW-UTSTATE		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-utstateCN
		UTIW-utstatePlacement

UMD Class Definitions for UTIW-USOR		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-usorCN

UMD Class Definitions for UTIW-USOR		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-usorPlacement

UMD Class Definitions for UTIW-UDOT		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-udotCN
		UTIW-udotPlacement

UMD Class Definitions for UTIW-USOE		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-usoeCN
		UTIW-usoePlacement

UMD Class Definitions for UTIW-Status		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-Status
		UTIW-DistList
		UTIW-PrimaryResTree

UMD Class Definitions for UTIW-Invisible		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-Invisible

UMD Class Definitions for UTIW-Public		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-PublicEmail

UMD Class Definitions for UTIW-Employee		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-Placement
		UTIW-AgencyNumber
		UTIW-LowOrg

UMD Class Definitions for UTIW-Employee		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-UserTemplate
		UTIW-InetMailDomain
		UTIW-PO
		UTIW-PODomain
		UTIW-NWAccess
		UTIW-EmpEmail
		UTIW-InsurEligible
		UTIW-LeaveEligible
		UTIW-BldgID

UMD Class Definitions for UTIW-CVV		
Inherits From	Mandatory Attributes	Optional Attributes
		UTIW-Placement
		UTIW-AgencyNumber
		UTIW-LowOrg
		UTIW-UserTemplate
		UTIW-InetMailDomain
		UTIW-PO
		UTIW-PODomain
		UTIW-NWAccess
		UTIW-EmpEmail
		UTIW-InsurEligible
		UTIW-LeaveEligible
		UTIW-BldgID

the power to chaNge™